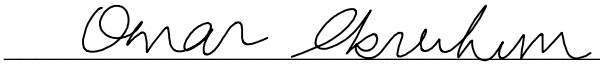
Omnigistics

by

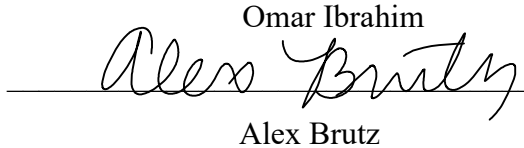
Omar Ibrahim, Elena Sanders & Alex Brutz

Submitted to
the Faculty of the School of Information Technology
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Technology

Elena Sanders — 4/26/2021
Date



Omar Ibrahim — 4/26/2021
Date



Alex Brutz — 4/26/2021
Date

Yahya Gilany

Yahya Gilany, Faculty Advisor

4/26/2021

Date

University of Cincinnati

College of

Education, Criminal Justice, and Human Services

April 2021

Contents

LIST OF ILLUSTRATION

ABSTRACT

Omnigistics is a scalable inventory management system that offers an intuitive approach to optimizing logistics operations for small to medium-sized companies. Small businesses either do not track inventory or use a manual method that can contribute to disorganization, resulting in costly inventory mistakes and accidental redundancy. Existing solutions offer inventory management modules as a smaller feature within an overcomplicated product. Omnigistics enables organizations to mass control item prices, combine goods and services, and track transactions. The application also improves customers' overall logistical process resulting in higher marginal profits and lower execution time.

1 INTRODUCTION

Introduction

Efficiency is crucial in the logistics industry and businesses across the supply chain use warehouses to store goods as they ship to customers whether they are consumers, retailers, manufacturers, or other business entities. “According to a study conducted by Wasp Barcode Technologies, 43% of small businesses either don’t track inventory or use a manual method” (Natarajan 2020).

We investigated four different highly rated solutions currently on the market, including Fishbowl, Odoo, Inflow and RedPrairie. For the first three applications, we have gone through them and identified a number of issues, such as redundant manual data input, a poor user interface, and a complicated workflow. We reached out to current users of the product for the review of RedPrairie. They mentioned the issues listed above, along with poor search functionality, and the long time it took to locate inventory items.

Problem

Based on this research, there is no solution that focuses solely on refining the organization's process by improving the common deficiencies identified in the area of inventory management. Organizations that use the tool will be able to manage their data, control item prices, combine goods and services, and track transactions.

Solution

Omnigistics is a secure, mobile friendly, easy-to-use web application that allows users to manage inventory and purchasing processes within the warehouse. It reduces input errors and provides better insight into how the business operates on a daily basis, as well as automating some manual tasks, such as maintaining stock levels. The Inventory Module will allow items to

be tracked separately or part of the composite kit along with price and quantity adjustments. The Purchasing Module will ensure that vendors receive purchase orders in a timely manner and track bills after the purchase of items.

Project Goals

The team's objective is to develop an inventory management system (IMS) as a software application that enables organizations to administer, control, and manage goods and services within the supply chain of their organization.

The project goals are to create the core of the application to manage the organizations data with the inventory and purchasing modules being the main features for managing the operational aspect.

The features of Omnigistics will include:

- User account management that includes employee profiles and access restrictions
- Handling different organizational data such as vendors, customers, and warehouses
- Managing different aspects of items within the warehouse which includes:
    - Items
    - Composite Items
    - Inventory Adjustments
    - Price Lists
- Managing the purchase orders, tracking bills, and restocking of items

Overview

This report describes the process that the Omnigistics team has undergone throughout the completion of the project. The following sections examine the process in detail: design objectives, methodology, budget, timeline, obstacles, and future recommendations.

2. DISCUSSION

Project Concept

The initial idea was based on Omar's previous work experience. While trying to research and implement a system for improving the workflow of the mechanic shop divisions of the company, it was hard to find a good system that met most requirements with a modern interface. While doing research into other types of systems that can be improved and different industries, he came across the logistics industry and inventory management systems. There was no perfect solution that met modern business requirements and the continuous growth of the industry so we decided to attempt to build something that could provide value for businesses within that sector.

Design Objectives

As we dive deeper into the project from the technology to the design and how we tackle the issue, the goal is continuously evolving. Initially, the objective was to develop a full-fledged warehouse management system with most, if not all, features from scratch. In view of the time we had, we realized that the project scope was too large, so we shifted focus to one part of the warehouse, which is inventory, the most crucial part.

The current goal is to include the main aspects of the inventory and some of its corresponding operations alongside supporting features such as purchasing. We chose to use and host the Microsoft technology stack consisting of ASP.NET Core, SQL, JavaScript, on the Azure cloud platform. This will allow for greater flexibility and sustainability in the long run, as there is a strong integration within the ecosystem.

Methodology and Technical Approach

The approach to the problem was to understand the business area from the perspective of workers and users. We talked to a professional in the industry and conducted research to better

understand the problem. The team started the development by identifying each feature and making small incremental changes until completion both on the front and back end. The design requirement was to create an intuitive and engaging interface to maximize user's satisfaction and increase productivity. The procedure that we followed to achieve this result was by developing both the front and the back end, and then integrating the two parts together to bring the application to life.

User Profiles

The following section comprises of three different user profiles representing various use cases for the Omnigistics application. User profiles focus on the collection of personal data linked to a specific user. This data is broken down into different categories such as experience needed, tasks, frequency of use, and design requirements. The following three user profiles below include logistics team employees or warehouse pickers, team leads or business owners, and software developers.

Figure 1: User Profile Form 1 *The following figure shows typical usage of the Omnigistics software by an employee or warehouse picker. These users would be the heavy users of the program accessing the site multiple times a day as needed. Those familiar with other warehouse and inventory management software or spreadsheet software can easily follow the workflow of Omnigistics.*

User Profile Form 1
Application: Omnigistics web application
Potential Users: Logistics team employees Warehouse picker
Software and Interface Experience: User should have experience launching a web browser and navigating to web portal to log in. Will need training in adding, deleting, combining, and managing items within application.

Experience with Similar Applications: Warehouse management software (Fishbowl, 3PL, NetSuite), Inventory management software (inFlow, Odoo), QuickBooks, Sage, Excel
Task Experience: User account creation, logging in with an email and password, inputting items into system, supervising warehouse stock, tracking bills.
Frequency of Use: Multiple times a day, whenever stock needs to be checked, or an order processed.
Key Interface Design Requirements that the Profile Suggests: The website needs to be intuitive, easy to learn, operate fast, and be reliable.

Figure I: User Profile Form 1

Figure 2: User Profile Form 2 *The following figure shows usage of the Omnigistics software by a team lead or the owner of a small business who will have access to an administrator account. An administrator account will allow the user to manage roles and permissions of other employees such as warehouse pickers in addition to supervising purchase orders and tracking bills. These users will need to be able to handle high level tasks including restocking and handling organizational data.*

User Profile Form 2
Application: Omnigistics web application
Potential Users: Logistics team lead Small business owners
Software and Interface Experience: User should have experience with basic features mentioned in Profile 1. In addition, user should have experience with controlling users, account creation, and role management
Experience with Similar Applications: Warehouse management software (Fishbowl, 3PL, NetSuite), Inventory management software (inFlow, Odoo), QuickBooks, Sage, Excel, Office 365 Portal, Active Directory
Task Experience: Tasks include supervising purchase orders, tracking bills, and restocking of items from a high level; setting up user account permissions; and handling different organizational data such as vendors, customers, and warehouses.
Frequency of Use: Multiple times a day, less frequent than a warehouse picker, may only need to log into as admin to manage roles
Key Interface Design Requirements that the Profile Suggests:

Role assignment must be clear so that the user understands what the various permission settings mean.

Figure II: User Profile Form 2

Figure 3: User Profile Form 3 *The following figure shows a user profile for a developer who will be maintaining the front end, database, and back-end code after initial launch. Developers will need to understand the various programming languages that were used to create the program. Adding new features and fixing bugs would be included in this user's workflow.*

User Profile Form 3
Application: Omnigistics MVC, Azure Devops, MS SQL, HTML/CSS, JavaScript, ASP.NET Core, C#, Azure
Potential Users: Developers of Omnigistics
Software and Interface Experience: Developers should be familiar with coding web applications and the MVC architecture. Database experience needed for those working on the back end. Experienced in front end design needed for changing visual aspects of application.
Experience with Similar Applications: Familiarity with programming languages used in application. Experience in using Git to manage code and MS SQL to manage database.
Task Experience: Tasks include maintaining website, creating new modules, using API to extend application features, and integrating with other applications.
Frequency of Use: Continuous development would occur after the official launch of the application. New features, bug fixes, and security vulnerabilities will be added at an on-release basis.
Key Interface Design Requirements that the Profile Suggests: API documentation needs to be clear and a clear development workflow must be established.

Figure III: User Profile Form 3

Use Case Diagram

All possible users and their corresponding system interactions are shown in the Use Case Diagram. Several actions that can be performed by different user types are detailed in both the front and back-end sections. Front-end users include team leads and warehouse workers as well as maintenance from web developers. In addition to maintenance and improvement by the

developers, front-end tasks primarily consist of use-case scenarios that clients can perform. Back end users will include Omnigistics employees mainly developers and some of their operations.

Figure 4: Use Case Diagram *The following diagram, Figure 4, demonstrates the use case for Omnigistics.*

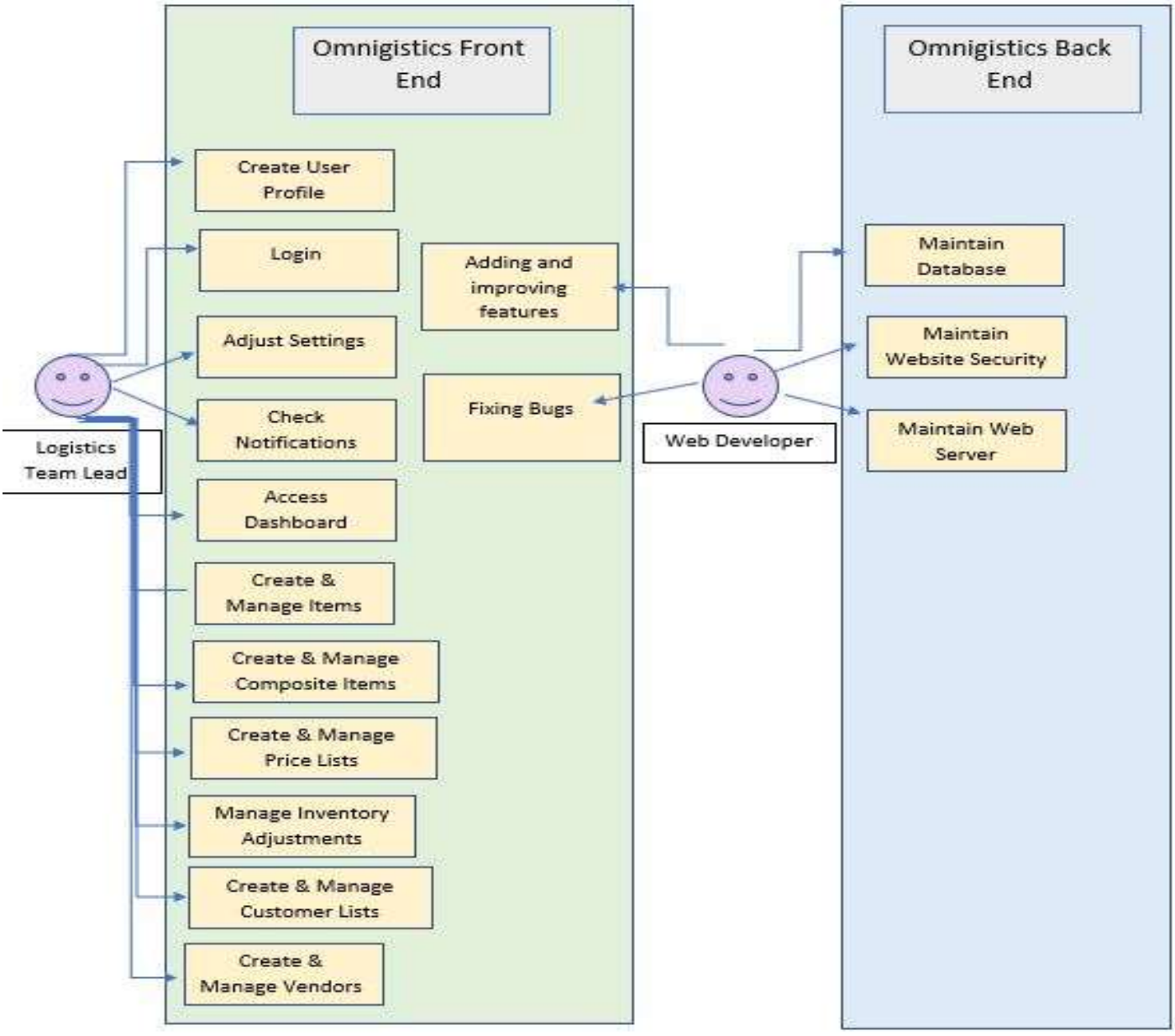


Figure IV: Use Case Diagram

Technical Architecture

Omnigistics was built in ASP.NET Core 3.1 using MVC alongside SQL server for data storage. Other notable technologies include JavaScript, AJAX, and Bootstrap. We also have a pipeline setup between our ADO repository and website which pulls the latest changes from the master branch and publishes a new version. For security we hash all sensitive information using SHA-256 and we use cookies for authentication.

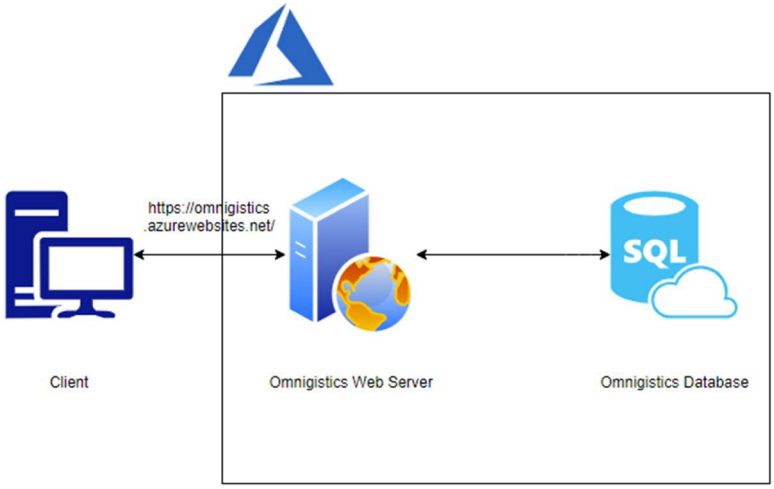


Figure V: Technical Architecture Diagram

Testing

The Omnigistics team will detail the methodologies used for testing and why we chose them, the scope of our tests, use case scenarios and reasons those were chosen features to test. Additionally, we will list the objectives of performing these tests and provide logs of our tests and procedures performed.

Testing Methodology

Our methodology for testing will consist of multiple tests to verify that the overall system operates as expected and to guarantee our focus group's optimal experience. The first type of test will include basic integration tests in order to ensure that the different pages and modules integrate as expected. The goal of doing this is to provide us with an overall analysis and view of

our system before we have users interact with it to minimize any potential problems that could have gotten past us. The second and primary focus will be user acceptance testing. In order to collect valuable feedback, we will bring together multiple peers and professionals to test our product. We will provide them with different use cases crucial to our product and monitor any failures or successes encountered during this test. We will also note any observations we make as the process is in progress and create surveys to gather feedback after the test is completed. Next, we will review our results and draw conclusions and the lessons learned throughout. Based on the feedback, we will make changes in order to make the user experience as positive as possible.

Scope of Testing

The scope of testing will include the core features consisting of inventory and purchasing modules within different roles in the system:

- Items: The core feature of our application is the ability to manage the various commodities stored within a warehouse. The main user will be the frontline employees on the warehouse managing the items A user should be able to create and manage all data points required to operate efficiently. The testing will ensure that all validation works as expected as well as ensuring a seamless experience for the person interacting with the system. Users should also be able to lookup items easily with the simple search lookup or using more advance searching features provided such as parameter filtering.
- Business data points: In order to operate the differing pages, basic information needs to be setup properly and maintained by employees. This includes things such as customers, vendors, warehouses, and price lists if available. The purpose is to provide the business with a central location to manage all datapoints required to run their inventory.

Employees within the organization should have proper access to maintain this data as the business grows and they should be able to easily keep up with the changes.

- Purchasing: As a complementary module to items, we want to ensure that all incoming items are tracked and managed properly. This module will allow the back office works to request item purchases from their vendors and send emails from within the system. This will also provide them with a log of all purchasing transactions performed and link to the billing portion which indicates a purchase has been completed. Within billing they should be able to track all payments made on the purchase orders and make sure all transactions are completed.
- Authorization: Admins should have access to the entire system and maintain the settings of the organization and different user privileges in addition to access levels by assigning different pre-defined user roles. This ensures the safety of the data and simplifies the workflow of the different users within the system.

Objectives

1. Each use case will represent a different user role within the system.
2. All complaints by testers will be investigated and concluded by the IT expo.
3. A test will be completed for all notable features in the product.
4. Results of user surveys yield positive metrics.
5. Identify and eliminate any defects brought to our attention during testing.

Test Logs and Procedures

Table 1: User Testing *The table below displays the user test results.*

USER	FEATURES TESTED

	Login	Create Account	Filter Table Results	View Dashboard	Navigate to Settings	Add Item	Create Purchase Order
RANDY SANDERS	X	X	X	X	X	X	X
SOPHIA RIZZI	X	X	X	X	X	X	X
HANNAH DAVEY	X	X	X	X	X	X	X
CECILIA BULGRIN	X	X	X	X	X	X	X

Table I: User Testing

Budget

The project budget displays the costs required to keep Omnigistics running after the initial development release. The budget reflects the operating costs if the application were to go into a commercial environment, as well as the costs for the maintenance of the implemented services. Services include a subscription to Visual Studio Professional which would be used as the development IDE, in addition to Microsoft Azure for application and database hosting. Finally, estimated labor costs for the application development were calculated.

Table 2: Project Budget *The table below displays the budget for the project. It is an estimation of real-world costs and reflects industry standard software.*

Omnigistics Budget				
NO.	ITEM	UNIT, HOURS	UNIT PRICE	TOTAL

SOFTWARE				
1	Visual Studio Professional	1	$1199 (yearly)	$1199
2	Azure App Service	1	$586.92 (yearly)	$586.92
3	Azure SQL Database	1	$606.72 (yearly)	$606.72
4	SendGrid Email API	1	$179.40 (yearly)	$179.40
	Subtotal			$2572.04
LABOR				
5		960	$30 (hourly)	$2880
				$2800
	Total			**$5452.04**

Table II: Project Budget

Project Timeline

The timeline is inclusive of both the fall semester and the spring semester. The project timeline has been updated as of the end of the fall semester. It includes dates for Analysis, Deliverables, Design, Development, Testing, and Implementation & Release. Please note that dates for the Spring semester 2021 are estimated. The timeline is concluded by the IT expo.

Table 3: Project Timeline *The table below displays the project schedule.*

Task #	Task Name	Duration (days)	Start Date	End Date
1.0	**Analysis**	**7**	**8/17/2020**	**8/24/2020**
1.1	Research Problem	2	8/17/2020	8/19/2020
1.1.1	Ideas and Brainstorming	1	8/18/2020	8/19/2020

1.1.2	Project Name	1	8/19/2020	8/20/2020
1.2	Research Hardware Solutions	1.5	8/21/2020	8/22/2020
1.3	Research Software Solutions	1.5	8/22/2020	8/23/2020
2.0	**Deliverables**	**69**	**8/19/2020**	**11/30/2020**
2.1	Team Members & Project Name	5	8/19/2020	8/24/2020
2.2	Team Contract	8	8/24/2020	9/1/2020
2.3	Project Abstract for Tech Expo	7	10/5/2020	10/12/2020
2.4	Team Contract Resubmission	7	10/5/2020	10/12/2020
2.5	User Profile	7	10/12/2020	10/19/2020
2.6	Use Case Diagram	7	10/12/2020	10/19/2020
2.7	Draft Report	7	11/2/2020	11/9/2020
2.8	Presentation and Demo	14	11/9/2020	11/23/2020
2.9	Final Fall Semester Report	21	11/9/2020	11/30/2020
3.0	**Design**	**9**	**9/1/2020**	**9/24/2020**
3.1	Design Database	3	9/1/2020	9/4/2020
3.2	Design Network Setup	3	9/4/2020	9/7/2020
3.3	Design API	3	9/7/2020	9/10/2020
3.4	Design UI	14	9/10/2020	9/24/2020
4.0	**Development**	**64**	**8/30/2020**	**12/14/2020**
4.1	Set up GitHub repo	1	8/30/2020	8/31/2020
4.2	Setup Database Server	2	9/24/2020	9/26/2020
4.3	Setup API environment	1	9/26/2020	9/27/2020
4.4	Setup UI environment	1	9/26/2020	9/27/2020
4.5	Create DB script	3	9/27/2020	9/30/2020
4.6	Develop API Endpoints Layout	7	9/30/2020	10/10/2020
4.7	Implement API Logic and integrate with DB	14	10/10/2020	10/24/2020
4.8	Develop UI Components	18	10/24/2020	11/11/2020
4.9	Implement State Management	7	11/12/2020	11/23/2020
4.10	Integrate UI with API endpoints	18	11/23/2020	12/14/2020
4.11	Purchasing module	30	1/18/2021	2/17/2021

4.12	Integration with purchasing module	14	2/18/2021	3/2/2021
4.13	Integration with shipping	14	3/3/2021	3/17/2021
5.0	**Testing**	**12**	**12/14/2020**	**12/22/2020**
5.1	Perform server testing	2	12/14/2020	12/16/2020
5.2	Performance testing	2	12/16/2020	12/18/2020
5.3	Responsive web UI testing	2	12/18/2020	12/20/2020
5.4	Responsive mobile UI testing	2	12/22/2020	12/24/2020
5.5	Security testing	2	12/24/2020	12/26/2020
5.6	Bug fixes	3	12/26/2020	12/28/2020
5.7	User testing	7	1/11/2021	1/18/2021
6.0	**Implementation& Release**	**12**	**3/31/2021**	**4/13/2021**
6.1	Create expo presentation	7	3/31/2021	4/7/2021
6.2	Create Omnigistics documentation	4	4/8/2021	4/12/2021
6.3	IT Expo	1	4/13/2021	4/13/2021

Table III: Project Timeline

Problems Encountered and Analysis of Problems Solved

Our general lack of coding experience was the first problem we encountered, we spent a much longer than we would have liked on learning the basics and working around simple error codes. Our solution was to find a middle ground on some of the technologies we used by incorporating different libraries and technologies that would simplify development and enable us to achieve the same outcome. It was easier to adapt to the various technologies introduced by utilizing the experience gained during research and debugging.

We also quickly realized that the original scope we had envisioned was too large to be reasonably addressed and implemented within the time allowed for this project. To counter this, we met and rethought the breadth of our idea. We revamped all the features and chose those that we felt should be highlighted in order to meet our goal of an inventory management system.

Finally, connecting the hosted database to the hosted application created its own problems as well. Microsoft Azure requires specific security settings and credentials needed to function. Our group had to overcome inexperience with additional research and trial and error. In the end, we completed this task and successfully hosted the application.

Future recommendations

If we were to do this project again, the main change would be firmly deciding on a technology stack earlier into the project timeline. Initially, we decided to build the application in React which is a JavaScript library for creating interactive user interfaces. This posed a challenge to those in the group who have not worked on enterprise development projects before. Spending less time recreating the application in ASP.NET would have led to the development of more features such as integration with a shipping platform or current online storefront.

After our first presentation, one suggestion we received was to make Omnigistics Americans with Disabilities Act (ADA) compliant. This would include adding text to speech for data entry, therefore removing the need to use a keyboard to add, edit, and manage inventory. While this was a great suggestion, time constraints and the scope of our project would not allow for implementing this. The idea of building an ADA compliant inventory management from the ground up would require a massive change in workflow and design.

3 CONCLUSION

Conclusion

This project has undergone many transformations during its development. The initial scope had to be minimized early on in order to have time to spend on a tangible product. In addition, a learning curve came with some of the required technology. As a result, new development and project management skills have been learned by the team. The project met its goals for the fall and spring semester successfully and most functionality within our scope has been delivered.

Lessons Learned

We underestimated the amount of time a task would take us on several occasions. This is because we were still adapting to the use of development technologies that we were unfamiliar with. Originally, we had a much larger scope, and in order to remain effective, we had to learn to focus our time. After trying to work on multiple features at once, we learned not to spread ourselves too thin. We sharpened our time management abilities to prioritize what needed to be done next, working on one feature at a time as opposed to many.

Abilities and Skills Developed Throughout Project

While creating this project, we had a big learning curve when it came to developing the front end of the application. The most important thing was time management and properly estimating time needed to complete the work. We learned new development skills using JavaScript, jQuery, Azure services, and Visual Studio. We also learned about the different process within project management as well as budgeting for a project and keeping costs in mind when utilizing technologies.

4 REFERENCES

Bustamante, F., Dekhne, A., Herrmann, J., & Singh, V. (2020, February 21). Improving warehouse operations--digitally. Retrieved September 01, 2020, from https://www.mckinsey.com/business-functions/operations/our-insights/improving-warehouse-operations-digitally

Natarajan, M. (2020, July 27). Common warehouse problems and solutions. Retrieved September 01, 2020, from https://www.zoho.com/inventory/guides/common-problems-in-warehouse-management-and-their-solutions.html

“The 4 essential components of a WMS.” Constantinou, M. (2020, June 15) Retrieved August 30, 2020, from https://www.zoho.com/creator/decode/the-4-essential-components-of-a wms/30, 2020

“What is a Warehouse Management System” (WMS) I Datex Corp. (n.d.). Retrieved August 30, 2020, from https://www.datexcorp.com/what-is-a-warehouse-management-system-wms/